

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Computing with formulas

### Foundation of programming (CK0030)

Francesco Corona

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## FdP

- ⊗ **Intro to variables, objects, modules, and text formatting**
- ⊗ Programming with WHILE- and FOR-loops, and lists
- ⊗ Functions and IF-ELSE tests
- ⊗ Data reading and writing
- ⊗ Error handling
- ⊗ Making modules
- ⊗ Arrays and array computing
- ⊗ Plotting curves and surfaces

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## A formula

### Computing with formulas

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## A formula

### Example

#### The vertical motion of a ball thrown up in the air

We can set up a mathematical model for the motion of the ball

- From Newton's second law of motion

The vertical position of the ball, called  $y$ , varies with time  $t$

$$y(t) = v_0 t - \frac{1}{2} g t^2 \quad (1)$$

↪  $v_0$  is the initial velocity of the ball

↪  $g$  is the acceleration of gravity

↪  $t$  is time

The  $y$  axis is chosen such that the ball starts from  $y = 0$  at  $t = 0$

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## A formula (cont.)

The time for the ball to move upwards and return to the ground again

We are interested in the solutions to equation  $y(t) = 0$

$$v_0 t - \frac{1}{2} g t^2 = t(v_0 - \frac{1}{2} g t) = 0 \quad (2)$$

$$\sim \begin{cases} t = 0 \\ t = 2 \frac{v_0}{g} \end{cases}$$

The ball returns to ground level in  $2v_0/g$  (seconds)

We can focus in the interval  $t \in [0, 2v_0/g]$

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## A formula (cont.)

$$y(t) = v_0 t - \frac{1}{2} g t^2$$

We evaluate the formula for some values of  $v_0$  and  $g$

- $v_0 = 5 \text{ ms}^{-1}$
- $g = 9.81 \text{ ms}^{-2}$

We want to compute the ball's height for  $t = 0.6 \text{ s}$

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

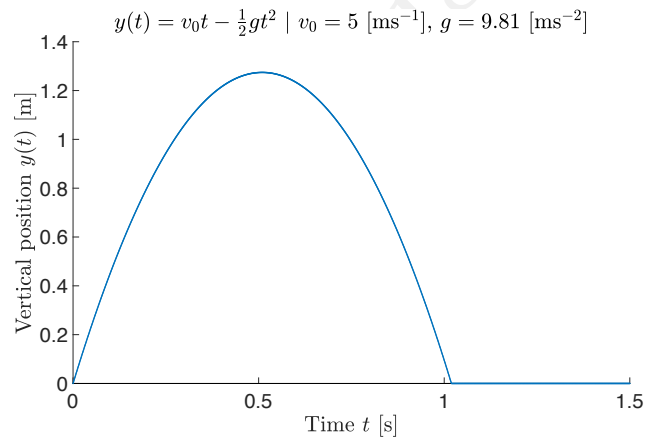
### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## A formula (cont.)



## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## A formula (cont.)

$$y = \underbrace{5}_{v_0} \cdot \underbrace{0.6}_t - \frac{1}{2} \cdot \underbrace{9.81}_g \cdot \underbrace{0.6^2}_{t^2} \quad (3)$$

```
1 print 5*0.6 - 0.5*9.81*0.6**2
```

### Remark

The four **standard arithmetic operators**

$\sim$  **+**, **-**, **\*** and **/**

Exponentiation employs a double asterisk **\*\*** notation

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## A formula (cont.)

The arithmetic expression is easily evaluated and printed

- A one-line Python program

The ball comes back after some time  $t = 2v_0/g \approx 1$  [s]

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Programs and programming

### A formula

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Programs and programming

Our task is to create programs/code and run it

There are three main types of tools for writing Python code

- A plain text editor
- An IPython notebook
- An integrated development environment (IDE) with a text editor

### Remark

What you choose depends on how you access Python

There are various possibilities

- Access a plain installation on your own computer
- Access a pre-installed environment (distribution)
- Access Python in a cloud service

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Programs and programming (cont.)

$$y(t) = v_0 t - \frac{1}{2} g t^2$$

$$\leadsto t = 0.6\text{s}$$

$$\leadsto v_0 = 5 \text{ ms}^{-1}$$

$$\leadsto g = 9.81 \text{ ms}^{-2}$$

```
1 print 5*0.6 - 0.5*9.81*0.6**2
```

This line is a **complete Python program** for evaluating the formula

- Copy the line in a **text file**
- Save the text file with name **ball1.py**

## Programs and programming (cont.)

The action required to run this program depends on the chosen tool

- Terminal window, IPython, Spyder, IPython notebook, ...

### Example

```
1 Terminal> python ball1.py
2
3 1.2342
```

After execution of `ball1.py`, the output (`1.2342`) is printed to screen

[Run me in a terminal, with/without ipython, in spyder, a notebook ... ]

## Programs and programming (cont.)

Suppose you now want to evaluate the formula for  $v_0 = 1$  and  $t = 0.1$

- 1 One must first edit the program text
- 2 Then, program must be re-executed

## Programs and programming (cont.)

### Example

$$y = \underbrace{1}_{v_0} \cdot \underbrace{0.1}_t - \frac{1}{2} \cdot \underbrace{9.81}_g \cdot \underbrace{0.1^2}_{t^2}$$

First edit the program text

```
1 print 1*0.1 - 0.5*9.81*0.1**2
```

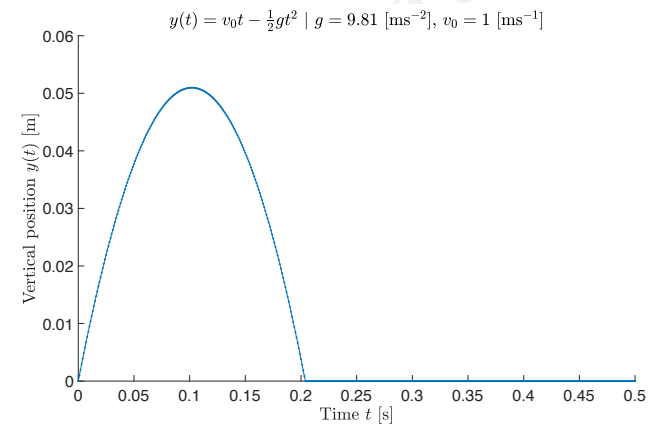
Then, re-execute the program

```
1 Terminal> python ball1.py
2
3 0.05095
```

The calculation has been changed

- The output is different
- `0.05095`

## A formula (cont.)



## Computing with formulas

FC  
CK0030  
2018.1

### A formula

#### Programs and programming

Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Programs and programming (cont.)

We had to modify the value of  $t$  at two places in our program

Every time we want to evaluate  $y(t)$  for different values of  $t$

$$y(t) = v_0 t - \frac{1}{2} g t^2$$

Such modifications could be made much simpler to perform

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

#### Programs and programming

Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Programs and programming (cont.)

We must express formulas in terms of symbols called **variables**

- Rather than numerical values

### Definition

#### Variables

In Python, **variables** are defined by setting a name (here  $v_0$ ,  $g$ ,  $t$ , or  $y$ ) equal to a numerical value or an expression involving already defined variables

Most programming languages, Python included, can use variables

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

#### Programs and programming

Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Programs and programming (cont.)

### Example

$$y(t) = v_0 t - \frac{1}{2} g t^2$$

```
1 v0 = 5
2 g = 9.81
3 t = 0.6
4
5 y = v0*t - 0.5*g*t**2
6
7 print y
```

This second program is much easier to read

- Closer to the mathematical notation

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

#### Programs and programming

Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Programs and programming (cont.)

↪ Store the program text in a file **ball2.py**

↪ Running the program outputs **1.2342**

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming

#### Variables

Comments, text and numbers formatting

### Another formula

Integer division

Objects in Python

Integer division

Arithmetic operators

### Mathematical functions

Examples

Rounding errors

### Interactive computing

The shell

Type conversion

IPython

# Variables and reserved words

A formula

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming

#### Variables

Comments, text and numbers formatting

### Another formula

Integer division

Objects in Python

Integer division

Arithmetic operators

### Mathematical functions

Examples

Rounding errors

### Interactive computing

The shell

Type conversion

IPython

## Variables and reserved words

Variable names can contain

- Any lower or upper case letter of the alphabet (A, a, B, b, ...)
- Numbers from 0 to 9 (but first character cannot be a number)
- Underscore (\_)

### Remark

Python distinguishes between upper and lower case letters

- Variable **X** is different from variable **x**
- Variable **Xx** is different from **xX**
- ...

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming

#### Variables

Comments, text and numbers formatting

### Another formula

Integer division

Objects in Python

Integer division

Arithmetic operators

### Mathematical functions

Examples

Rounding errors

### Interactive computing

The shell

Type conversion

IPython

## Variables and reserved words (cont.)

### Example

$$y(t) = v_0 t - \frac{1}{2} g t^2$$

```
1 initial_velocity = 5
2 acceleration_of_gravity = 9.81
3 TIME = 0.6
4
5 VerticalPositionOfBall = initial_velocity*TIME - \
6     0.5*acceleration_of_gravity*TIME**2
7
8 print VerticalPositionOfBall
```

With long variables names, the code for evaluating the formula got long

- We broke it into two lines (the backslash \ at the end of the line)
- Make sure there are no blanks after the backslash

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming

#### Variables

Comments, text and numbers formatting

### Another formula

Integer division

Objects in Python

Integer division

Arithmetic operators

### Mathematical functions

Examples

Rounding errors

### Interactive computing

The shell

Type conversion

IPython

## Variables and reserved words (cont.)

Long names explain well what they represent

Though checking correctness of the formula for **y** became harder

- (Than in the program using **v0**, **g**, **t**, and **y0**)

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming

#### Variables

Comments, text and numbers formatting

### Another formula

Integer division

Objects in Python

Integer division

Arithmetic operators

### Mathematical functions

Examples

Rounding errors

### Interactive computing

The shell

Type conversion

IPython

## Variables and reserved words (cont.)

A standard convention is to have variable names with lower case letters

- (Then, words are separated by an underscore)

### Example

Whenever the variable represents a mathematical symbol, we use it

- $v_0$  in mathematics becomes `v0` in the program
- $y$  in mathematics becomes `y` in the program

Resemblance between math symbols and variables names is important

- ~ Easy reading of the code
- ~ Errors detection

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming

#### Variables

Comments, text and numbers formatting

### Another formula

Integer division

Objects in Python

Integer division

Arithmetic operators

### Mathematical functions

Examples

Rounding errors

### Interactive computing

The shell

Type conversion

IPython

## Variables and reserved words (cont.)

Certain words are reserved in Python

- Utilised to build the language

These **reserved words** CAN NOT be used as variable names

- **and, as, assert, break, class, continue, def, del, elif, else, except, False, finally, for, from, global, if, import, in, is, lambda, None, nonlocal, not, or, pass, raise, return, True, try, with, while, yield**

### Remark

To use a reserved word as variable name, add an underscore at the end

- For some quantity  $\lambda$ , use `lambda_`

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming

#### Variables

Comments, text and numbers formatting

### Another formula

Integer division

Objects in Python

Integer division

Arithmetic operators

### Mathematical functions

Examples

Rounding errors

### Interactive computing

The shell

Type conversion

IPython

## Variables and reserved words (cont.)

Program files can have a freely chosen name

It is good practice to AVOID names coinciding with **keywords** or **module**

Keywords and module names in Python

- `math.py, time.py, random.py, os.py, sys.py`
- `while.py, for.py, if.py, class.py, def.py`
- ...

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming

#### Variables

Comments, text and numbers formatting

### Another formula

Integer division

Objects in Python

Integer division

Arithmetic operators

### Mathematical functions

Examples

Rounding errors

### Interactive computing

The shell

Type conversion

IPython

**Comments, text and number  
formats**  
A formula

## Comments, text and number formats

Along with code statements, it is always informative to provide **comments**

- To explain the idea behind statements
- Using a natural language

### Definition

#### Comments

In Python, **comments** start with the **#** character

Everything after **#** on a line is ignored when the program is executed

## Comments, text and number formats (cont.)

### Example

```
1 # Compute the height of a ball in vertical motion
2
3 v0 = 5           # initial velocity
4 g = 9.81         # acceleration of gravity
5 t = 0.6          # time
6
7 y = v0*t - 0.5*g*t**2 # vertical position
8
9 print y
```

## Comments, text and number formats (cont.)

### Remark

By default, **non-English characters** in comments are disabled

- If you use them, Python will complain

```
1 SyntaxError: Non-ASCII character '\xc3' in file ...
2 but no encoding declared; see
3 http://www.python.org/peps/pep-0263.html for details
```

Non-English characters are enabled by using a code line in the beginning

```
1 # -*- coding: utf-8 -*-
```

- This is a comment that is not ignored by Python

## Comments, text and number formats (cont.)

As output of our program we simply print a numerical value of *y*

```
1 # Compute the height of a ball in vertical motion
2
3 v0 = 5           # initial velocity
4 g = 9.81         # acceleration of gravity
5 t = 0.6          # time
6
7 y = v0*t - 0.5*g*t**2 # vertical position
8
9 print y
```

It is often a good idea to write/print a more informative text

For example, consider printing the example text

~ At t=0.6 s, the height of the ball is 1.23 m



## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables

Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Comments, text and number formats (cont.)

### Definition

**Printf syntax** (from function **printf** in the C programming language)

Output from a **print** statement, plus number formatting

The oldest and most widely used technique is **printf formatting/syntax**

- The **printf syntax** is used in a lot of other programming languages
- It is easy to learn and very convenient and flexible to work with
- The **syntax** of **printf formatting** may look awkward

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables

Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Comments, text and number formats (cont.)

The **print** statement prints a **string**

```
1 # Compute the height of a ball in vertical motion
2
3 v0 = 5           # initial velocity
4 g = 9.81         # acceleration of gravity
5 t = 0.6          # time
6
7 y = v0*t - 0.5*g*t**2 # vertical position
8
9 print 'At t=%g s, the height of the ball is %.2f m.' % (t, y) # print y
```

Everything enclosed in quotes (single, ' , or double " ) denotes a string

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables

Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Comments, text and number formats (cont.)

```
1 print 'At t=%g s, the height of the ball is %.2f m.' % (t, y)
```

The string above (based on our program) is formatted using **printf syntax**

- The string has '**slots**', starting with a percentage sign
- Variables in the program can be inserted in the slots

The slots and the variables in the example

~ %g and %.2f

~ t and g

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables

Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Comments, text and number formats (cont.)

```
1 print 'At t=%g s, the height of the ball is %.2f m.' % (t, y)
```

We have two '**slots**', thus two variables must be inserted into the slots

The relevant syntax is to list the variables inside parentheses after the string

- The variables' list is separated from it by a percentage symbol

~ % (t, y)

The first variable, **t**, goes into the first '**slot**' with format specification **%g**

- The percentage sign **marks the slot**
- The following character, **g**, is the chosen **format specification**
- The **g** format instructs the real number to be compactly written

The next variable, **y**, goes into the second '**slot**' with format **.2f**

- The **.2f** format instructs the real number is with two decimal digits
- ~ (The **f** in the **.2f** format stands for *floating-point number*)

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Comments, text and number formats (cont.)

### Example

```
1 v0 = 5
2 g = 9.81
3 t = 0.6
4
5 y = v0*t - 0.5*g*t**2
6
7 print 'At t=%g s, the height of the ball is %.2f m.' % (t, y)
```

```
1 Terminal> python ball_print_f.py
2
3 At t=0.6 s, the height of the ball is 1.23 m
```

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Comments, text and number formats (cont.)

There are many available ways to specify formats

**e** writes a number in **scientific notation**

- A number between 1 and 10 followed by a power of 10
- $(1.2432 \cdot 10^{-3})$ , as **1.2432e-03**
- Capital **E** in the exponent is possible: Replace **e** by **E** (**1.2432E-03**)

For **decimal notation** we use letter **f** (as in **%f**)

- The output number appears with digits before and/or after a comma
- (**0.0012432** instead of **1.2432E-03**)

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Comments, text and number formats (cont.)

With the **g** format, the output is in scientific notation for large or small numbers and it is in decimal notation otherwise (**compact output**)

- A lower case **g** leads to lower case **e** in scientific notation
- An upper case **G** implies **E** instead of **e** in the exponent

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Comments, text and number formats (cont.)

It is possible to specify the format in some very sophisticated manner

### Example

~> **10.4f**  
~> **14.6E**

The first case: A float written in decimal notation

- 4 decimals in a field of width equal to 10 characters

The second case: A float written in scientific notation

- 6 decimals in a field of width equal to 14 characters

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming

Variables

Comments, text and numbers formatting

### Another formula

Integer division

Objects in Python

Integer division

Arithmetic operators

Mathematical functions

Examples

Rounding errors

Interactive computing

The shell

Type conversion

IPython

## Comments, text and number formats (cont.)

| Format | Explanation   |
|--------|---|
| %s     | A string  |
| %d     | An integer  |
| %0xd   | An integer in a <b>x</b> -width field, padded with leading zeros  |
| %f     | Decimal notation with six decimals                                |
| %e     | Compact scientific notation, <b>e</b> in the exponent             |
| %E     | Compact scientific notation, <b>E</b> in the exponent             |
| %g     | Compact decimal or scientific notation, with <b>e</b>             |
| %G     | Compact decimal or scientific notation, with <b>E</b>             |
| %xz    | Format <b>z</b> right-adjusted in a <b>x</b> -width field         |
| %-xz   | Format <b>z</b> left-adjusted in a <b>x</b> -width field          |
| %.yz   | Format <b>z</b> with <b>y</b> decimals                            |
| %x.yz  | Format <b>z</b> with <b>y</b> decimals in a <b>x</b> -width field |
| %%     | The percentage sign   |

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming

Variables

Comments, text and numbers formatting

### Another formula

Integer division

Objects in Python

Integer division

Arithmetic operators

Mathematical functions

Examples

Rounding errors

Interactive computing

The shell

Type conversion

IPython

## Comments, text and number formats (cont.)

### Example

```

1 i = 62
2 r = 189876545.7654675432
3
4 # Print out numbers with quotes " to see width of field
5
6 print '%d' % i          # minimum field
7 print '%5d' % i         # field of width 5 characters
8 print '%05d' % i        # pad with zeros
9
10 print '%g' % r          # r is big number, scientific notation
11 print '%G' % r          # E in the exponent
12 print '%e' % r          # compact scientific notation
13 print '%E' % r          # compact scientific notation
14 print '%20.2E' % r      # 2 decimals, field of width 20
15 print '%30g' % r        # field of width 30 (right-adjusted)
16 print '%-30g' % r       # left-adjust number
17 print '%-30.4g' % r     # 3 decimals
18
19 print '%s' % i           # convert i to string automatically
20 print '%s' % r
21
22 # Use %% to print the percentage sign
23 print '%g %% of %.2f Euro is %.2f Euro' % \
24      (5.1, 346, 5.1/100*346)

```

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming

Variables

Comments, text and numbers formatting

### Another formula

Integer division

Objects in Python

Integer division

Arithmetic operators

Mathematical functions

Examples

Rounding errors

Interactive computing

The shell

Type conversion

IPython

## Comments, text and number formats (cont.)

### Example

```

1 v0 = 5
2 g = 9.81
3 t = 0.6
4
5 y = v0*t - 0.5*g*t**2
6
7 print """
8 At t=%f s, a ball with
9 initial velocity v0=%.3E m/s
10 is located at the height %.2f m.
11 """ % (t, v0, y)

```

A **triple-quoted string**, started and ended by three single/double quotes

Triple-quoted strings are used for text that spans several lines

- **t** is printed in the **f** format (by default six decimals)
- **v0** is written in the **.3E** format (three decimals and the number spans as narrow field as possible)
- **y** is two decimals in narrow decimal notation, **.2f**

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming

Variables

Comments, text and numbers formatting

### Another formula

Integer division

Objects in Python

Integer division

Arithmetic operators

Mathematical functions

Examples

Rounding errors

Interactive computing

The shell

Type conversion

IPython

## Comments, text and number formats (cont.)

```

1 Terminal> python ball_print2.py
2
3 At t=0.600000 s, a ball with
4 initial velocity v0=5.000E+00 m/s
5 is located at the height 1.23 m.

```

- **t** is printed in the **f** format (by default six decimals)
- **v0** is written in the **.3E** format (three decimals and the number spans as narrow field as possible)
- **y** is two decimals in narrow decimal notation, **.2f**

## Comments, text and number formats (cont.)

### Format string syntax

It offers all the functionalities available with the printf format

- (And much more, through a different syntax)

### Example

We illustrate this syntax on the one-line output that was used earlier

```
1 print 'At t={t:g} s, the height of the ball is {y:.2f} m.' \
2     .format(t=t, y=y)
```

- Slots are denoted by curly braces (rather than a percentage sign)
- Variable are listed with an optional colon and format specifier
- Variables and their values are listed at the end of the statement
- Slots have names (the sequence of variables is not important)

## Comments, text and number formats (cont.)

At times, we want to write out text that spans several lines

```
1 print """
2 At t={t:f} s, a ball with
3 initial velocity v0={v0:.3E} m/s
4 is located at the height {y:.2f} m.
5 """.format(t=t, v0=v0, y=y)
```

We can obtain such an output by using triple-quoted strings

## Comments, text and number formats (cont.)

### The newline character

We can also use ordinary single-quoted strings and a special character

- The special character indicates where line breaks should occur
- The special character is `\n` (a backslash followed by letter `n`)

### Example

```
1 print """y(t) is
2 the position of
3 our ball."""
4
5 print 'y(t) is \nthe position of\nour ball'
```

The two `print` statements have identical output

```
1 y(t) is
2 the position of
3 our ball.
```

Another formula  
Computing with formulas

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Another formula

### Example

Consider the usual expression for converting a temperature measurement

- From degrees Celsius ( $C$ ) to its value in degrees Fahrenheit ( $F$ )

$$F = \frac{9}{5}C + 32 \quad (4)$$

Given the formula above and a value of  $C$ , our goal is to compute  $F$

A first attempt at implementing the formula

```
1 C = 21
2 F = (9/5)*C + 32
3 print F
```

The parentheses are not strictly needed

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Another formula (cont.)

$$F = \frac{9}{5}C + 32$$

```
1 C = 21
2 F = (9/5)*C + 32
3 print F
```

When run under Python version 2.x, the program prints the value 53

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Another formula (cont.)

$$F = \frac{9}{5}C + 32$$

Testing correctness is easy, we evaluate the formula on a calculator

$$\sim \frac{9}{5} \cdot 21 + 32 = 69.8 \neq 53$$

What is wrong? The formula typed in the program looks correct!

```
1 C = 21
2 F = (9/5)*C + 32
3 print F
```

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

**Integer division**  
Another formula

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Integer division

The error is one of the most common errors in mathematical coding

- For a newcomer to programming, this is not obvious at all

In many computer languages, there are two types of divisions

- **Integer division**
- **Float division**

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Integer division

### Definition

**Float division** is what you expect from standard arithmetics

- $9/5$  becomes 1.8 in decimal notation

**Integer division**  $a/b$  with integers  $a$  and  $b$  is an integer  $c$

- It is the largest integer  $c$  such that  $bc \leq a$

$\leadsto 9/5$  is 1, as  $1 \cdot 5 = 5 \leq 9$  and  $2 \cdot 5 = 10 > 9$

$\leadsto 1/5$  is 0, as  $0 \cdot 5 \leq 1$  and  $1 \cdot 5 > 1$

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Integer division (cont.)

### Remark

Many computer languages (... , Fortran, C, C++, Java, and Python 2.x) interpret  $a/b$  as integer division, if both operands  $a$  and  $b$  are integers

Suppose that either  $a$  or  $b$  are real (floating-point) numbers

- $\leadsto$  Then,  $a/b$  implies the standard mathematical division
- $\leadsto$  (Float division)

Other languages (... , MATLAB and Python 3.x) interpret  $a/b$  as float division even if both operands are integers

- (or complex division if any of the operands is a complex number)

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Integer division (cont.)

The issue with our program is in the coding of the formula  $(9/5)*C + 32$

```
1 C = 21
2 F = (9/5)*C + 32
3 print F
```

First,  $9/5$  is calculated (Python interprets 9 and 5 as integers)

- $9/5$  is thus interpreted as a division between two integers
- Python chooses by default integer division, returning 1

Then, 1 is (normally) multiplied by C, giving 21

- 21 and 32 are added
- 53 is returned

$\leadsto$  (Wrong result)

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
**Objects in Python**  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

# Objects in Python

Another formula

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
**Objects in Python**  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Objects in Python

Consider a very general assignment statement like `C = 21`

- Python interprets number `21` as an integer

```
1 C = 21
```

It creates an **int** (for integer) **object** holding the value `21`

- The variable `C` acts as **variable name**
- This name labels the **int object** as `C`

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
**Objects in Python**  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Objects in Python (cont.)

Similarly, in `C = 21.0`, Python recognises number `21.0` as a real number

```
1 C = 21.0
```

It creates a **float** (for floating-point) **object** holding the value `21.0`

- The variable `C` is the **variable name** of this **float object**

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
**Objects in Python**  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Objects in Python (cont.)

### Remark

The key issue is that `21` and `21.0` are identical numbers in mathematics

In Python,

- `21` gives an **int object**
- `21.0` gives a **float object**

## A formula

Programs and  
programming  
Variables  
Comments, text and  
numbers formatting

## Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

Mathematical  
functions

Examples  
Rounding errors

Interactive  
computing

The shell  
Type conversion  
IPython

## Objects in Python (cont.)

## Remark

Any (Python) assignment statement has the general form

- Variable name, on the left-hand side
- An object, on the right-hand side

```
1 C = 21
1 v0 = 5
2 g = 9.81
3 t = 0.6
4
5 y = v0*t - 0.5*g*t**2
```

## A formula

Programs and  
programming  
Variables  
Comments, text and  
numbers formatting

## Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

Mathematical  
functions

Examples  
Rounding errors

Interactive  
computing

The shell  
Type conversion  
IPython

## Objects in Python (cont.)

At this point, it is not requested to know now what an object exactly is

As initial simplification, one can think of an `int` object as a collection

- It is like a storage box, with some information about an integer
- The information is stored within the computer's memory
- The variable name `C` is used to access this information

There are various object types, some are pre-built some are user-defined

- Objects may contain a lot of data, not just integer/real numbers

## A formula

Programs and  
programming  
Variables  
Comments, text and  
numbers formatting

## Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

Mathematical  
functions

Examples  
Rounding errors

Interactive  
computing

The shell  
Type conversion  
IPython

## Objects in Python (cont.)

## Example

```
1 print 'A text with an integer %d and a float %f' % (2, 2.0)
```

A `str` (for string) object, without a name, is first created from 'the text between quotes' and then the `str` object is printed using `print` command

We can alternatively do this in two, sequential, steps

```
1 s = 'A text with an integer %d and a float %f' % (2, 2.0)
2 print s
```

## A formula

Programs and  
programming  
Variables  
Comments, text and  
numbers formatting

## Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

Mathematical  
functions

Examples  
Rounding errors

Interactive  
computing

The shell  
Type conversion  
IPython

Avoiding integer division  
Another formula



## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Avoiding integer division

We must be careful to avoid integer division when coding math formulas

When a program is asked to implement an integer division

- A double forward-slash (`//`) should be used as division operator

This is Python's way of explicitly indicating integer division

### Remark

Python 3.x has no problem with *unintended* integer division

- Only with Python 2.x (and other languages)

There are several ways to avoid integer division with the plain (`/`) operator

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Avoiding integer division (cont.)

The simplest remedy in Python version 2

```
1 from __future__ import division
```

This import statement must be present in the beginning of EVERY single file where the `/` operator ALWAYS shall imply float division

An alternative remedy, one can run any Python program `someprogram.py` from the command line with the argument `-Qnew` for the Python interpreter

```
1 Terminal> python -Qnew someprogram.py
```

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Avoiding integer division (cont.)

A more widely used method, common also to other programming languages

↪ Force one of the operands to be a `float object`

### Example

```
1 F = (9.0/5)*C + 32
2 F = (9/5.0)*C + 32
3
4 F = float(C)*9/5 + 32
```

In the first two lines, one of the operands is written as a decimal number

- This implies a `float object`, and therefore float division results

In the last line, `float(C)*9` means (`float times int`)

- This results in a `float object`, and thus float division is implicit

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Avoiding integer division (cont.)

### Example

```
1 F = C*float(9/5) + 32 # !! It does not work correctly !!
```

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Avoiding integer division (cont.)

We can ask our Python to locate *potential* integer divisions in a program

- Python programs can be executed with a `-Qwarnall` argument

It will show a warning every time an integer division expression is found

```
1 Terminal> python -Qwarnall someprogram.py
```

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Avoiding integer division (cont.)

### Remark

We could have run into problems when we wrote the formula  $\frac{1}{2}gt^2$

- We used `0.5*g*t**2`, and that worked well
- If `(1/2)*g*t**2`, term `(1/2)` would be zero

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Arithmetic operators

Another formula

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Arithmetic operators

In Python, formulas are evaluated as they are mathematically

- Given an expression, from left to right, term by term
- The terms are separated by plus (+) or minus (-)

Within terms, power operations ( $a^b$ , `a**b`) have priority

- Computed before multiplication/division

Parentheses dictate how a formula is evaluated

## Arithmetic operators (cont.)

### Example

1 `5/9 + 2*a**4/2`

First, `5/9` (`5/9`) is evaluated (as integer division, with `0` as result)

`a4` (`a**4`) is evaluated, `2` and `a4` are multiplied (`2*a**4`)

- The result is divided by `2` (`2*a**4/2`)
- The answer is therefore `a**4`

This result is added to the result of `5/9` (`5/9 + a**4`)

## Arithmetic operators (cont.)

### Example

1 `5/(9+2)*a**(4/2)`

First, expression  $\frac{5}{9+2}$  ( $\sim 5/(9+2)$ ) is evaluated (integer division, result `0`)

`4/2` (`4/2`) is computed (integer division, result `2`)

`a**2` (`a**(4/2)`) is calculated

This result is multiplied by the result of `5/(9+2)` (`5/(9+2)*a**(4/2)`)

The answer is thus always `0`

## Arithmetic operators (cont.)

It is easy to unintentionally get integer division in formulas

Of course, it is possible to turn integer division off in (any) Python

- Important to be aware of the existence of the concept
- Important to develop programming habits to avoid it

### Remark

The concept of integer division appears in many programming languages

- It is better to learn as early as possible how to deal with it
- Python-specific (or else) features does not remove the issue

Mathematical functions  
Computing with formulas

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Evaluating mathematical functions

Standard mathematical formulas frequently involve common functions

- `sin`, `cos`, `tan`, `sinh`, `cosh`, `exp`, `log`, ...

On a pocket calculator you have special buttons for such functions

- Similarly, in a language you have ready-made *functions*

### Remark

In principle, one could write his/her own program for evaluating

- Say, the `sin(x)` function

How to do it efficiently is often a non-trivial task

Experts have worked on such problem for decades

- They implemented their best recipes
- These codes should be (are) re-used

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Evaluating mathematical functions (cont.)

We discuss how to reach `sin`, `cos`, and similar functions within Python

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Examples Mathematical functions

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Examples, sqrt and sinh

### Example

The height *y* of a ball in vertical motion, with initial upward velocity *v*<sub>0</sub>

$$y = v_0 t - \frac{1}{2} g t^2$$

In the formula, we are using *g* for the gravity acceleration and *t* for time

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

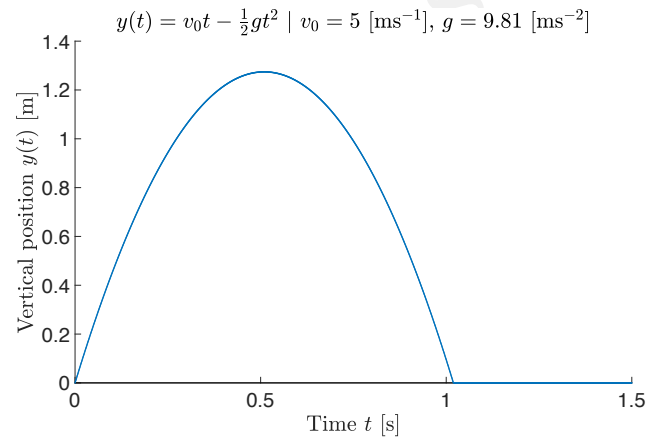
**Examples**  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Examples, sqrt and sinh (cont.)

How long time does it take for the ball to reach the height  $y_c$ ?



There are two solutions ( $t_1$  and  $t_2$ )

- Once when the ball reaches  $y_c$  on its way up ( $t_1$ )
- Once when it reaches on its way down ( $t_2$ )

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

**Examples**  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Examples, sqrt and sinh (cont.)

- When  $y = y_c$ , we have  $y_c = v_0 t - \frac{1}{2} g t^2$  and the equation

$$\frac{1}{2} g t^2 - v_0 t + y_c = 0 \quad (5)$$

- A quadratic form<sup>1</sup> that must be solved with respect to  $t$

$$t_1 = \frac{v_0 - \sqrt{v_0^2 - 2g y_c}}{g}$$

$$t_2 = \frac{v_0 + \sqrt{v_0^2 - 2g y_c}}{g}$$

For the expressions of  $t_1$  and  $t_2$ , we need the square root  $[\sqrt{\cdot}]$

$$^1 a x^2 + b x + c = 0, x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}.$$

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

**Examples**  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Examples, sqrt and sinh (cont.)

### Remark

Square root and other math functions are available in a **module** called **math**

- sin, cos, sinh, exp, log, ...

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

**Examples**  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Examples, sqrt and sinh (cont.)

To make module functions available, we must first **import the module**

- We must write **import math** in our program

To take the square root of variable  $a$ ,  $\sqrt{a}$ , we write **math.sqrt(a)**

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Examples, sqrt and sinh (cont.)

### Example

$$t_{(1|2)} = \frac{v_0 \mp \sqrt{v_0^2 - 2gy_c}}{g}$$

```
1 v0 = 5
2 g = 9.81
3 yc = 0.2
4
5 import math
6 t1 = (v0 - math.sqrt(v0**2 - 2*g*yc))/g
7 t2 = (v0 + math.sqrt(v0**2 - 2*g*yc))/g
8
9 print 'At time t=%g s and %g s, the height is %g m.' % (t1, t2, yc)
```

The output from this program

```
1 At time t=0.0417064 s and 0.977662 s, the height is 0.2 m.
```

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Examples, sqrt and sinh (cont.)

### Definition

The standard way to import a module, *module\_name*

~ *import module\_name*

Functions *function\_name* are accessed by using *module\_name* as prefix

~ *module\_name.function\_name*

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Examples, sqrt and sinh (cont.)

### Example

```
1 import math
2 x = math.sqrt(y)
```

Clearly, the use of `math.sqrt(y)` is less pleasing than a plain `sqrt(y)`

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Examples, sqrt and sinh (cont.)

### Definition

An alternative import syntax allows to skip the module name prefix

- *from module\_name import function\_name*

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Examples, sqrt and sinh (cont.)

A specific example of `from module_name import function_name`

~ `from math import sqrt exp log sin`

The alternative import syntax allows direct access to `sqrt` (or else)

- Without the `math.` prefix

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Examples, sqrt and sinh (cont.)

### Example

$$t_{(1|2)} = \frac{v_0 \mp \sqrt{v_0^2 - 2gy_c}}{g}$$

```
1 v0 = 5
2 g = 9.81
3 yc = 0.2
4
5 from math import sqrt
6 t1 = (v0 - sqrt(v0**2 - 2*g*yc))/g
7 t2 = (v0 + sqrt(v0**2 - 2*g*yc))/g
```

# WAS: import math

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Examples, sqrt and sinh (cont.)

### Definition

All functions *function\_name* in module *module\_name* can be imported at once

~ `from module_name import *`

- Importing all (\*) functions from a module is often convenient
- Not recommended to import more functions than needed
- The convenience of a compact import syntax often wins

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Examples, sqrt and sinh (cont.)

In the `math` module

- `sin`, `cos`, `tan`, `asin`, `acos`, `atan`, `sinh`, `cosh`, `tanh`
- `exp`, `log` (base *e*), `log10` (base 10), `sqrt`
- Numbers (`e`, `pi`, ...)
- ...

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Examples, sqrt and sinh (cont.)

### Definition

Modules and functions can be given new names in the import statement

### Example

```
1 import math as m                # m is now the name of the math module
2
3 v = m.sin(m.pi)
4
5 from math import log as ln
6 v = ln(5)
7
8 from math import sin as s, cos as c, log as ln
9 v = s(5)*c(5) + ln(5)
```

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Examples, sqrt and sinh (cont.)

### Remark

Since in Python everything is an object

Modules, functions, numbers and strings are objects

Variables refer to objects and new variables may refer to them

```
1 m = math
2 ln = m.log
3 s = m.sin
4 c = m.cos
```

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

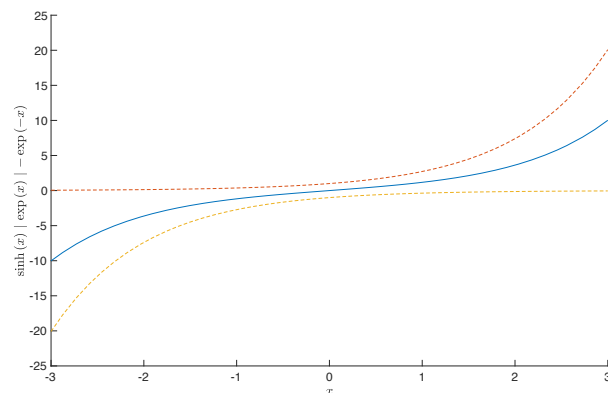
The shell  
Type conversion  
IPython

## Examples, sqrt and sinh (cont.)

### Example

Consider the definition of the hyperbolic function  $\sinh(x)$

$$\sinh(x) = \frac{1}{2}(e^x - e^{-x}) \quad (6)$$



## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Examples, sqrt and sinh (cont.)

$$\sinh(x) = \frac{1}{2}(e^x - e^{-x})$$

We can evaluate  $\sinh(x)$  in three different ways

- By calling `math.sinh`, directly
- By computing the RHS using `math.exp`
- By computing the RHS using `e` and power expressions `math.e**x` and `math.e**(-x)`



## A formula

Programs and  
programming  
Variables  
Comments, text and  
numbers formatting

## Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

Mathematical  
functions

## Examples

Rounding errors

Interactive  
computing

The shell  
Type conversion  
IPython

## Examples, sqrt and sinh (cont.)

## Example

$$\sinh(x) = \frac{1}{2}(e^x - e^{-x}), \text{ for } x = 2\pi$$

```
1 from math import sinh, exp, e, pi
2
3 x = 2*pi
4
5 r1 = sinh(x)
6 r2 = 0.5*(exp(x) - exp(-x))
7 r3 = 0.5*(e**x - e**(-x))
8
9 print r1, r2, r3
```

All three computations are mathematically equivalent

- Output from `print` displays identical results

```
1 267.744894041 267.744894041 267.744894041
```

..., SQN!

## A formula

Programs and  
programming  
Variables  
Comments, text and  
numbers formatting

## Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

Mathematical  
functions

## Examples

## Rounding errors

Interactive  
computing

The shell  
Type conversion  
IPython

## Rounding errors

### Mathematical functions

## A formula

Programs and  
programming  
Variables  
Comments, text and  
numbers formatting

## Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

Mathematical  
functions

## Examples

## Rounding errors

Interactive  
computing

The shell  
Type conversion  
IPython

## Rounding errors

## Example

```
1 from math import sinh, exp, e, pi
2
3 x = 2*pi
4
5 r1 = sinh(x)
6 r2 = 0.5*(exp(x) - exp(-x))
7 r3 = 0.5*(e**x - e**(-x))
8
9 print '%.16f %.16f %.16f' % (r1,r2,r3)      # WAS: print r1, r2, r3
```

A print out of `r1`, `r2`, `r3` that displays 16 decimals

```
1 267.7448940410164369
2 267.7448940410164369
3 267.7448940410163232
```

This command shows how `r1`, `r2`, `r3` are different

- But, why is it so?

## A formula

Programs and  
programming  
Variables  
Comments, text and  
numbers formatting

## Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

Mathematical  
functions

## Examples

## Rounding errors

Interactive  
computing

The shell  
Type conversion  
IPython

## Rounding errors (cont.)

## Remark

A computer program calculates its arithmetics using *wannabe* real numbers<sup>2</sup>

True real numbers (Dedekind) may require an infinite number of decimals

- ~ Because of finite storage, the sequence of decimals is truncated
- ~ On computers, it is standard to keep 17 digits in a real number

<sup>2</sup>Let  $x \in \mathcal{R}$  and let  $\text{fl}(x)$  its (rounded) representation in a computer. We have that  $x \neq \text{fl}(x)$  with  $\frac{|x - \text{fl}(x)|}{|x|} \leq \frac{1}{2}\varepsilon_M$  in which the quantity  $\varepsilon_M$  is called *machine precision*.

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Rounding errors (cont.)

### Remark

Real numbers on a computer often have a small error

Only a few numbers can be represented exactly

- The rest are approximations

Most arithmetic operations on a computer involve inaccurate real numbers

- This results in inaccurate calculations

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Rounding errors (cont.)

### Example

Think of  $\frac{1}{49} = 1$  and  $\frac{1}{51} = 1$  when performed in Python

```
1 print '%.16f %.16f' % (1/49.0*49, 1/51.0*51)
```

```
1 0.9999999999999999 1.0000000000000000
```

- $1/49$  is not correctly represented in the computer

- $1/51$  also has an inexact representation

~ (but error does not show too much :/)

Errors in floating-point numbers may propagate through computations

The results are approximations to the exact mathematical values

- Such errors are commonly called **rounding errors**

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Rounding errors (cont.)

### Remark

Python has ad hoc modules **decimal** and **SymPy** package has module **mpmath**

They allow for real numbers to be represented with adjustable accuracy

- Rounding errors can be made as small as desired

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

**Interactive computing**  
Computing with formulas

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Interactive computing

Python can execute statements and evaluate expressions interactively

The environments where one works interactively are Python **shells**

- The simplest Python 2.x shell is invoked by **python** or **python2**

~ (In a terminal)

```
1 Terminal> python
2 Python 2.7.9 (default, Jun 29 2016, 13:08:31)
3 [GCC 4.9.2] on linux2
4 Type "help", "copyright", "credits" or "license" for more information.
5
6 >>>
```

Some Python messages are displayed together with a **prompt** `> > >`

- After that, you can start issuing commands

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Interactive computing (cont.)

### Example

#### The interactive shell as calculator

- 1 Type **3\*4.5-0.5**
- 2 Press **Return**

```
1 Terminal> python
2 Python 2.7.9 (default, Jun 29 2016, 13:08:31)
3 [GCC 4.9.2] on linux2
4 Type "help", "copyright", "credits" or "license" for more information.
5
6 >>> 3*4.5-0.5
7      13.0
```

The text after the `> > >` prompt is the **shell input**

The text without the `> > >` prompt is the result that Python calculates

- The **shell output**

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## Interactive computing (cont.)

### Remark

The shell makes it easy to recover previous input and edit the text

~ This helps experimenting with statements and expressions

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

**The shell**  
**Interactive computing**

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

#### The shell

Type conversion  
IPython

## The shell

### Example

Consider the program for the vertical position of the ball

```
1 v0 = 5
2 g = 9.81
3 t = 0.6
4 y = v0*t - 0.5*g*t**2
5 print y
```

It can be fully re-typed line-by-line in the Python shell

```
1 >>> v0 = 5
2 >>> g = 9.81
3 >>> t = 0.6
4 >>> y = v0*t - 0.5*g*t**2
5 >>> print y
6 1.2342
```

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

#### The shell

Type conversion  
IPython

## The shell (cont.)

We can easily calculate the **y** value corresponding to another **v0** value

- Hit the arrow-up key (↑), to recover previous statements
- Repeat pressing ↑, until the **v0 = 5** statement shows up
- You can then edit the relative line

```
1 >>> v0 = 6 # It was: v0 = 5
```

- Press **Return**, to execute this statement
- To check the new value of **v0** either type **v0** or **print v0**

```
1 >>> v0
2 6
3
4 >>> print v0
5 6
6
7 >>> print y # Old, needs be re-computed
8 1.2342
```

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

#### The shell

Type conversion  
IPython

## The shell (cont.)

The next step is to re-compute **y**, with the new **v0** value

- Hit the arrow-up key (↑) multiple times to recover the statement where **y** is assigned
- Press **Return**
- Write **y** or **print y** to see the result

```
1 >>> y = v0*t - 0.5*g*t**2
2 >>> y
3 1.8341999999999996
4
5 >>> print y
6 1.8342
```

We get two slightly different results

- **y** prints out all the decimals stored in the computer (16)
- **print y** prints out **y** with fewer decimals, standard format

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

#### The shell

Type conversion  
IPython

## The shell (cont.)

### Remark

Computations on a computer often suffer from rounding errors

- The present calculation is no exception

The correct answer is **1.8342**

Rounding errors led to a number that is incorrect

- The error is in the 16th decimal
- The error is  $4 \cdot 10^{-16}$

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell

### Type conversion

IPython

# Type conversion

## Interactive computing

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell

### Type conversion

IPython

## Type conversion

Work w/o bothering about the **type of objects** variables refer to

- Yet, we encountered a serious problem with integer division
- Important to be careful about the involved types of objects

The interactive shell is useful for exploring types (the **type** function)

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell

### Type conversion

IPython

## Type conversion (cont.)

### Example

Let us create some **int** object **C**

Let us check its type with **type(C)**

~ Function **type**

```
1 >>> C = 21
2 >>> type(C)
3 <type 'int'>
4
5 >>> C
6 21
```

We convert the **int** object **C** to a corresponding **float** object

~ Function **float**

```
1 >>> C = float(C) # type conversion
2 >>> type(C)
3 <type 'float'>
4
5 >>> C
6 21.0
```

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell

### Type conversion

IPython

## Type conversion (cont.)

Statement **C = float(C)** creates a new object, **C**

~ From the original one, **C**

The new object is also referred to by the name **C**

~ It binds it to the same name **C**

After the statement, variable **C** refers to a different object

- Original **int** object, value **21**, becomes unreachable

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell

### Type conversion

IPython

## Type conversion (cont.)

### Example

We can also convert a **float** object to a corresponding **int** object

~ Function **int**

```
1 >>> C = 20.9
2 >>> type(C)
3 <type 'float'>
4
5 >>> D = int(C)
6 >>> type(D)
7 <type 'int'>
8
9 >>> D
10 20
# type conversion
# decimals are truncated
```

Converting a **float** to an **int** implied stripping off the decimals

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell

### Type conversion

IPython

## Type conversion (cont.)

### Example

Conversion according to rounding rules

~ Function **round**

```
1 >>> round(20.9)
2 21.0
3
4 >>> int(round(20.9))
5 21
```

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell

### Type conversion

IPython

## IPython Interactive computing

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell

### Type conversion

IPython

## IPython

There are several improvements of the standard Python shell

- **IPython** is the common interactive shell
- You need to have **ipython** installed

Typing **ipython** in a terminal window starts the shell

```
1 Terminal> ipython
2 Python 2.7.9 (default, Jun 29 2016, 13:08:31)
3 Type "copyright", "credits" or "license" for more information.
4
5 IPython 2.3.0 -- An enhanced Interactive Python.
6 ? -> Introduction and overview of IPython's features.
7 %quickref -> Quick reference.
8 help -> Python's own help system.
9 object? -> Details about 'object', use 'object??' for extra
10 details.
11
12 In [1]:
```

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## IPython (cont.)

### Definition

The (default) prompt in *ipython* is not `> > >` but `In [X]:`

- *X* is the number of the present input command

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## iPython (cont.)

### Example

#### Running programs

```
1 Terminal> ipython
2 Python 2.7.9 (default, Jun 29 2016, 13:08:31)
3 Type "copyright", "credits" or "license" for more information.
4
5 IPython 2.3.0 -- An enhanced Interactive Python.
6 ?                -> Introduction and overview of IPython's features.
7 %quickref        -> Quick reference.
8 help            -> Python's own help system.
9 object?         -> Details about 'object', use 'object??' for extra
10                  details.
11
12 In [1]: run ball2.py
13         1.2342
```

The command requires that you have `cd`'ed to the folder with `ball2.py`

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## IPython (cont.)

On Windows you may, as alternative to starting IPython from a DOS or PowerShell window, double click on the IPython icon or use Start menu

- You must move to the folder where your program is located
- If `ball2.py` is in the folder `div` under `My Stuff` of user `me`
- (This is done by the `os.chdir`, change directory, command)

```
1 In [1]: import os
2 In [2]: os.chdir(r'C:\Documents and Settings\me\My Stuff\div')
3 In [3]: run ball2.py
```

- Note the `r` before the quote in the string
- Required to let a backslash (`\`) really mean the backslash character

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## IPython (cont.)

### Remark

You may frequently have to type the `os.chdir` command in *ipython*

This and other commands can be suitably placed in a *startup file*

~ A file that is automatically executed when you launch *ipython*

~ To create one from Terminal, *ipython create profile*

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## IPython (cont.)

Inside **ipython** you can invoke any operating system command

This allows to navigate the filesystem with Unix/Windows commands (**cd**)

- (Instead of Python's, **os.chdir**)

```
1 In [1]: cd C:\Documents and Settings\me\My Stuff\div
2 In [3]: run ball2.py
```

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## IPython (cont.)

### Remark

Suppose that some Python variables have the same name as an OS command

- (**date=30**)

The OS command must be called with an exclamation mark (!) in front

- (**!date**)

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## IPython (cont.)

It is recommended to run all Python programs from inside **ipython**

- **ipython** can help examine the state of variables and locate bugs
- (When something goes wrong)

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## IPython (cont.)

### Remark

- ~ To execute a program in **ipython**, type **run** before program name
- ~ To run a program in a Terminal, **python** prior to program name



## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## IPython (cont.)

### Definition

Output from statements or expressions in *ipython* are preceded by *Out [X]*

- *X* is the command number of the last *In [X]* prompt

When programs are executed, as with the *run* command or when OS commands are run, the output is from the OS itself

- In this case, the output is not preceded by any *Out [X]* label

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## IPython (cont.)

### Definition

#### Output recovery

Outputs (*Out [X]*) from previous statements in *ipython* are available/usable

They are in variables of the form *\_iX* (underscore *\_*, *i*, and a number *X*)

*X* is *1* for the last statement, *2* for the second last statement, and so forth

- Short forms are *\_* (for *\_i1*), *--* (for *\_i2*), and *---* (for *\_i3*)

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## IPython (cont.)

### Example

Remember that the output from input *In [1]* was *1.2342*

We can now refer to it by an underscore

We can also perform operations on it

- Say, we multiply it by *10*

```
1 In [2]: _*10
2 Out[2]: 12.341999999999999
```

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## IPython (cont.)

### Definition

#### Command recovery

The command history is navigated by typing *Ctrl+p*

- *↑* for going backward
- *Ctrl+n* or *↓* for going forward

Any command you hit can be edited and re-executed

- Also commands from previous sessions
- (They are in the history)

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## IPython (cont.)

### Definition

#### Command history

The **command history** from previous **ipython** sessions is available

- This feature makes it easy to modify work from a previous session
- Hit arrow-up to recall commands and edit them as needed

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## IPython (cont.)

### Definition

#### Tab completion

Pressing the **TAB** key completes incompletely typed variable names

- It can save some typing

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## IPython (cont.)

### Definition

#### OS commands

```
1 In [3]: date
2         Thu Nov 18 11:06:16 CET 2010
3
4 In [4]: ls
5         myfile.py yourprog.py
6
7 In [5]: mkdir mytestdir
8
9 In [6]: cd mytestdir
```

## Computing with formulas

FC  
CK0030  
2018.1

### A formula

Programs and programming  
Variables  
Comments, text and numbers formatting

### Another formula

Integer division  
Objects in Python  
Integer division  
Arithmetic operators

### Mathematical functions

Examples  
Rounding errors

### Interactive computing

The shell  
Type conversion  
IPython

## IPython (cont.)

### Remark

#### Notebooks

Alternative to interactive shells

It allows to record/replay interactive sessions as a mix

- Text, mathematics, Python code, and graphics